

A PRELIMINARY REPORT ON

**GENERATING RELEVANT QUESTIONS FOR
A QUERY USING NATURAL LANGUAGE
PROCESSING TECHNIQUES.**

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN THE FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

OF

**BACHELOR OF ENGINEERING
(COMPUTER ENGINEERING)**

SUBMITTED BY

HRUSHABH HIRUDKAR
ANUJ KANETKAR
SHRINIWAS NAYAK

Exam No : 71700888G
Exam No : 71700929H
Exam No : 71701044K



DEPARTMENT OF COMPUTER ENGINEERING

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

DHANKAWADI, PUNE - 411043



CERTIFICATE

This is to certify that the Project Entitled

Project Title

Submitted by

HRUSHABH HIRUDKAR
ANUJ KANETKAR
SHRINIWAS NAYAK

Exam No : 71700888G
Exam No : 71700929H
Exam No : 71701044K

are bonafide students of this institute and the work has been carried out by them under the supervision of **Dr. A. S. Ghotkar**, and it is approved for the fulfillment of the requirements of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

Dr. A. S. Ghotkar
Guide,
Dept. of Computer Engg.

Prof. Mrs. M. S. Takaliker
Head,
Dept. of Computer Engg.

Dr. P. T. Kulkarni
Principal,
Pune Institute of Computer Technology

Place: Pune

Date:

Acknowledgements

It gives us great pleasure in presenting the preliminary project report on ‘**Generating relevant questions for a query using natural language processing techniques**’.

We would like to take this opportunity to thank our internal guide **Dr. A. S. Ghotkar** for giving us all the help and guidance we needed. We are really grateful to them for their kind support. Their valuable suggestions were very helpful.

We are also grateful to **Prof. Mrs. M. S. Takalikar**, Head of Computer Engineering Department, PICT for her indispensable support, suggestions.

In the end our special thanks to **Ms Jugnu Manhas** and **Mr Nikhil Malhotra** for providing valuable insights for our Project. It was our pleasure to work in the Maker’s Lab, Tech Mahindra.

We would also like to thank our family and friends, without their support this would not have been possible.

Hrushabh Hirudkar
Anuj Kanetkar
Shriniwas Nayak
(B.E. Computer Engg.)

Abstract

This report focuses on the development of a system to generate relevant questions for a query using Natural Language Processing. In today's technologically advancing world, getting answers for questions, general or specific is germane to the development of the user and as a result, development of the overall community. In this report, we propose to find the relevant questions to a query entered by the user to solve doubts. Using text similarity, we can find out relevant questions to the query put forth by the user. We also propose to generate new relevant questions to the query entered by the user.

Keywords:

NLP, Information Retrieval, Lexical Similarity, Semantic Similarity, String Similarity, Corpus Similarity, Text Generation, NLU, NLG

Contents

List of Abbreviations	i
List of Figures	ii
List of Tables	iii
1 INTRODUCTION	1
1.1 MOTIVATION	2
1.2 PROBLEM DEFINITION	2
2 LITERATURE SURVEY	3
3 SOFTWARE REQUIREMENTS SPECIFICATION	5
3.1 INTRODUCTION	6
3.1.1 PROJECT SCOPE	6
3.1.2 USER CLASSES AND CHARACTERISTICS	7
3.1.3 ASSUMPTIONS AND DEPENDENCIES	8
3.2 FUNCTIONAL REQUIREMENTS	9
3.3 EXTERNAL INTERFACE REQUIREMENTS	10
3.3.1 USER INTERFACES	10

3.3.2	COMMUNICATION INTERFACES	11
3.4	NON-FUNCTIONAL REQUIREMENTS	11
3.4.1	PERFORMANCE REQUIREMENTS	11
3.4.2	SAFETY REQUIREMENTS	11
3.4.3	SECURITY REQUIREMENTS	11
3.4.4	SOFTWARE QUALITY ATTRIBUTES	12
3.5	SYSTEM REQUIREMENTS	12
3.6	SDLC MODEL	13
3.7	SYSTEM IMPLEMENTATION PLAN	15
4	SYSTEM DESIGN	16
4.1	SYSTEM ARCHITECTURE	17
4.1.1	BIRD'S EYE VIEW	18
4.1.2	NLU DETAILED ARCHITECTURE	19
4.2	DATA FLOW DIAGRAMS	20
4.2.1	DFD LEVEL 0	20
4.2.2	DFD LEVEL 1	21
4.3	UML DIAGRAMS	22
4.3.1	USE CASE DIAGRAM	22
5	OTHER SPECIFICATIONS	23
5.1	ADVANTAGES	24
5.2	LIMITATIONS	24
5.3	APPLICATIONS	24

6	BASIC IMPLEMENTATION AND RESULT	25
6.1	ALGORITHM	26
6.2	RESULT	27
7	CONCLUSION AND FUTURE SCOPE	30
7.1	CONCLUSION	31
7.2	FUTURE SCOPE	31
	Appendix A Feasibility Study	32
A.1	PROBLEM STATEMENT FEASIBILITY	33
	Appendix B Details of Papers	34
B.1	Details Of Papers	35
	Appendix C Plagiarism Report	36
8	REFERENCES	40

List of Abbreviations

ABBREVIATION	ILLUSTRATION
NLP	Natural Language Processing
NLU	Natural Language Understanding
NLG	Natural Language Generation
SDLC	Software Development Life Cycle
NLTK	Natural Language Tool Kit
UI	User Interface
UML	Unified Modeling Language
GCP	Google Cloud Platform
GPU	Graphics Processing Unit
JSON	JavaScript Object Notation
P	Polynomial Time Problem
NPC	Non deterministic Polynomial Time Complete Problem
NPH	Non deterministic Polynomial Time Hard Problem

List of Figures

3.1	Class Diagram	8
3.2	Agile Model	13
4.1	Basic Architecture - Overview	18
4.2	NLU Detailed Architecture	19
4.3	DFD Level 0	20
4.4	DFD Level 1	21
4.5	Use Case Diagram	22
6.1	Suggested Questions	27
6.2	JSON File	28
6.3	Frequency Count of Similarity Index	29
C.1	Plagiarism Report 1	37
C.2	Plagiarism Report 2	38
C.3	Plagiarism Report 3	39

List of Tables

2.1 Literature Survey	4
---------------------------------	---

CHAPTER 1

INTRODUCTION

According to stack exchange the parent site of stack overflow boasts of 10 million views daily. All these views are for catering to their immediate requirement which stack overflow is available to fulfill but fails to develop the overall concept of the user.

1.1 MOTIVATION

In today's technologically advancing world, getting answers for questions, general or specific is very important to the development of the user and as a result, development of the overall community eventually. Thus, we propose to build a system to find the relevant questions to a query entered by the user to solve doubts thereby enhancing knowledge of the user. Using text similarity, we can find out relevant questions to the query put forth by the user.

One of the most popular sources for people to find answers to their questions in StackOverflow. On StackOverflow people can easily find answers to the questions they have, but there is no suggestion for more questions on StackOverflow. We only get the answers to the question which we have asked, but if more similar questions are suggested this would generate inquisitiveness in the user regarding the subject. This would ultimately help in increasing the knowledge of users by going through relevant question chain.

1.2 PROBLEM DEFINITION

Many technology enthusiasts including professionals working in the industry, researchers and students from all over the world use stack overflow as a ready reckon er for almost any difficulty they face in the technical field, while this proves extremely beneficial at the moment but does not help to develop the concept as a whole. Use of stack overflow solves the problem temporarily but does not help in enhancing knowledge.

CHAPTER 2

LITERATURE SURVEY

TITLE	DESCRIPTION	DATASET USED	RESULT
Unsupervised Sparse Vector Densification for Short Text Similarity - Song et al (2015)	Combining ESA and word2vec for better similarity measure for shorter text	20 news group dataset for short text similarity	Combining yielded better result than individual operation
Computing text similarity using Tree Edit Distance - Sidorov et al (2015)	TED for similarity between n-grams	Simple English Sentences	Gives better accuracy as compared to Levenchstein's SED
Robust semantic text similarity using LSA, machine learning, and linguistic resources - Kashyap et al (2015)	Term alignment algorithm implementation	Simple paragraphs of literature essays	Best system for sentence-phrase, phrase-word similarity
A Survey of Text Similarity Approaches- International Journal of Computer Applications (0975 – 8887) Volume 68– No.13, April 2013	Text Similarity Approaches		Study about various Text Similarity Approaches such as Character Based, Corpus Based, Knowledge Based
Corpus-based and Knowledge-based Measures of Text Semantic Similarity	Approaches for measuring semantic similarity.		Detailed Study about Corpus based and Knowledge Similarity.
Natural Language Generation in the context of the Semantic Web	Natural Language Generation in Semantic context.		Semantic based study and Natural Language Generation.

Table 2.1: Literature Survey

CHAPTER 3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 INTRODUCTION

The system to be built aims at providing relevant questions to an input query or question from a database and also generate questions/queries that may prove to be beneficial to the user. The system can be used across many sectors and will prove beneficial in learning, improving profits and reducing time of exploratory tasks by providing help to the user.

The system must satisfy basic functional as well as non-functional requirements as discussed in subsequent text in order to qualify as a useful and appreciable project. Any project or system in today's world has to be secure, user friendly and fast enough to survive in the market or to produce itself as a viable option.

3.1.1 PROJECT SCOPE

This project is currently being built with stackoverflow database but the system will be able to produce desired when fed with a similar database consisting of set of questions from any field. The project not only aims at understanding the fed query and presenting similar questions from the dataset but also aims at generating questions that will help the user to understand the subject matter at hand better.

The project scope pans mainly over presenting and generating queries based on an input user query when presented with a database. The project however does not extend to answering these questions or suggesting any resource to study the same. The project consists of two stages namely NLU (Natural Language Understanding) and NLG (Natural Language Generation).

3.1.2 USER CLASSES AND CHARACTERISTICS

This sub-section discusses the role and privilege level enjoyed by different users of the system. The users of the system can be identified into these broad categories as follows :

- Front End User
- Back End User

The front-end user will be exposed only to the front UI of the system and will be able to query the system and provide feedback based on the output of the system namely the suggested or generated questions. The feedback intended from the front user is predominantly based on the user's judgement of the relevance or usefulness regarding the queries suggested/generated. The Backend user can again be split as follows:

- Admin level user
- System level user

The system level user will be able to upload and modify the database available with the system whereas the admin level user will be able to analyse the performance of the system with the help of the feedback in addition to the functionalities of system level user.

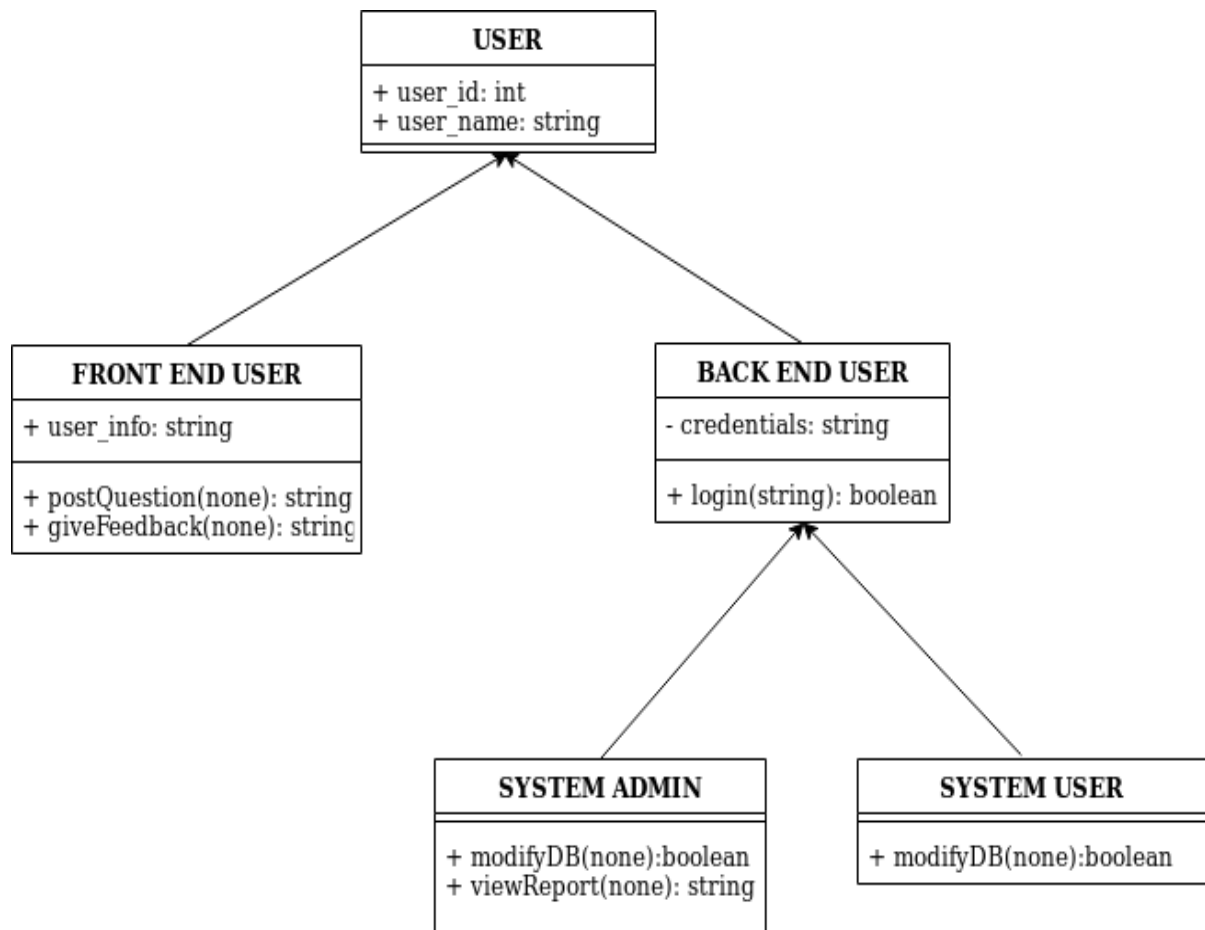


Figure 3.1: Class Diagram

3.1.3 ASSUMPTIONS AND DEPENDENCIES

This subsection will discuss the implicit as well as explicit assumptions that can be made without the loss of generality with respect to usage of the system in varied fields. The system assumes that the data set presented is coherent and does not include any contradiction in the text. Also, the presented data set will contain appreciable data so that the system can produce cogent output.

The system assumes that the complete data is in English. The system is supposed to work on English NLP, the scope may be enhanced to Indic NLP in future.

3.2 FUNCTIONAL REQUIREMENTS

Functional requirements are used to define a function of the system or a component of the system as a relationship between input and output. Functional requirements for the proposed system have been discussed in this section. All functional mainly revolve around these three points:

- Purpose - To generate relevant questions for the entered query
- Input - A query or a question which is to be entered by the user
- Output - Set(s) of questions generated for the entered query

The functional requirements for the system can be stated as follows:

- Suggest relevant Questions from the database as per the input query which pass some similarity threshold, also check the similarity threshold for the generated questions.
- Record the feedback of the user and store the same efficiently for analysis in future. The input query can also be retained as the part of the data set.
- Generate reports about the performance of the system and send the same periodically to the system manager or admin as the requirement may be

The above requirements sum up the main functional requirements of the system. Addition in this chapter is possible as the scope of the system increases.

3.3 EXTERNAL INTERFACE REQUIREMENTS

This section discusses the need of external Interface that will be implemented in the system. External interface requirements are usually define under four main subsections namely user, hardware, software and communication.

3.3.1 USER INTERFACES

In today's world user interface serves not any as a utility for users to interact with the system but also as a selling point for many applications across different domains and fields of work. The user must find the interface comfortable and navigation in the system should be intuitive and as per the general standards. The UI must provide functionality to the user in accordance with the level of access he/she has.

The UI must have the following characteristics:

- Comfortable to use
- Lucrative Design
- Easy to modify
- Intuitive to navigate
- Attractive to end user

The system will mainly be used by front end and back end user, the UI must provide functionality for back end users to operate on database and view the performance of the system.

Currently the system does not require any hardware or software interfaces, the same can be included later if need there be.

3.3.2 COMMUNICATION INTERFACES

Communication interfaces include email, messaging or network protocols that will be required by the system either to fulfill the requirement or provide a functionality. The system under consideration may be required to send periodical mails or files to system admin containing the reports regarding the functioning of the system.

For this system communications interface mainly contains reporting of the system performance reports to the system admin. The reports can be generated in JSON format which happens to easy to use, transport and work on.

3.4 NON-FUNCTIONAL REQUIREMENTS

Non-Functional Requirement (NFR) defines the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to success of the software system.

3.4.1 PERFORMANCE REQUIREMENTS

The system should be able to give a quick and efficient response for the entered query. The delay of response, if any, should be only due to lack of computation power and not due to inefficiency of the algorithm

3.4.2 SAFETY REQUIREMENTS

- System should not crash due to overabundance of queries
- Abrupt failure of system should not happen

3.4.3 SECURITY REQUIREMENTS

- System should maintain a threshold of queries to be handled simultaneously so as to avoid getting into aborted state

- Addition of question to the database must be available for only those queries entered by the user i.e. direct addition to the database without displaying similar questions should not happen

3.4.4 SOFTWARE QUALITY ATTRIBUTES

The software quality attributes for the system under consideration are :

- Maintainability - Need of updation of database whenever query is entered
- Robust - The software should be able to handle multiple queries at a time
- Recoverability - The database should have replicas in case of loss of data
- Availability - User should be able to query the database anytime

3.5 SYSTEM REQUIREMENTS

This section deals with the requirements for the working of the system like Database Requirement, Software Requirements and Hardware Requirements.

- Database Requirements - The database is a BigQuery database, so the Google Cloud Platform (GCP) should be enabled for querying the database if required.
- Hardware Requirements
 - Intel i5 Processor
 - GPU for performing computations on large datasets
- Software Requirements
 - NLTK Library
 - Python 3

3.6 SDLC MODEL

SDLC stands for Software Development of Life Cycle and represents the methodology for efficient implementation that will be followed during completing the entire project. Different models like Waterfall, V-Model, Agile methodology were taken into consideration and Agile methodology has been chosen for the project as it best suits the nature of the project.

Agile model has six phases namely Planning, Analysis, Design, Implementation, Testing and Integration and Maintenance.

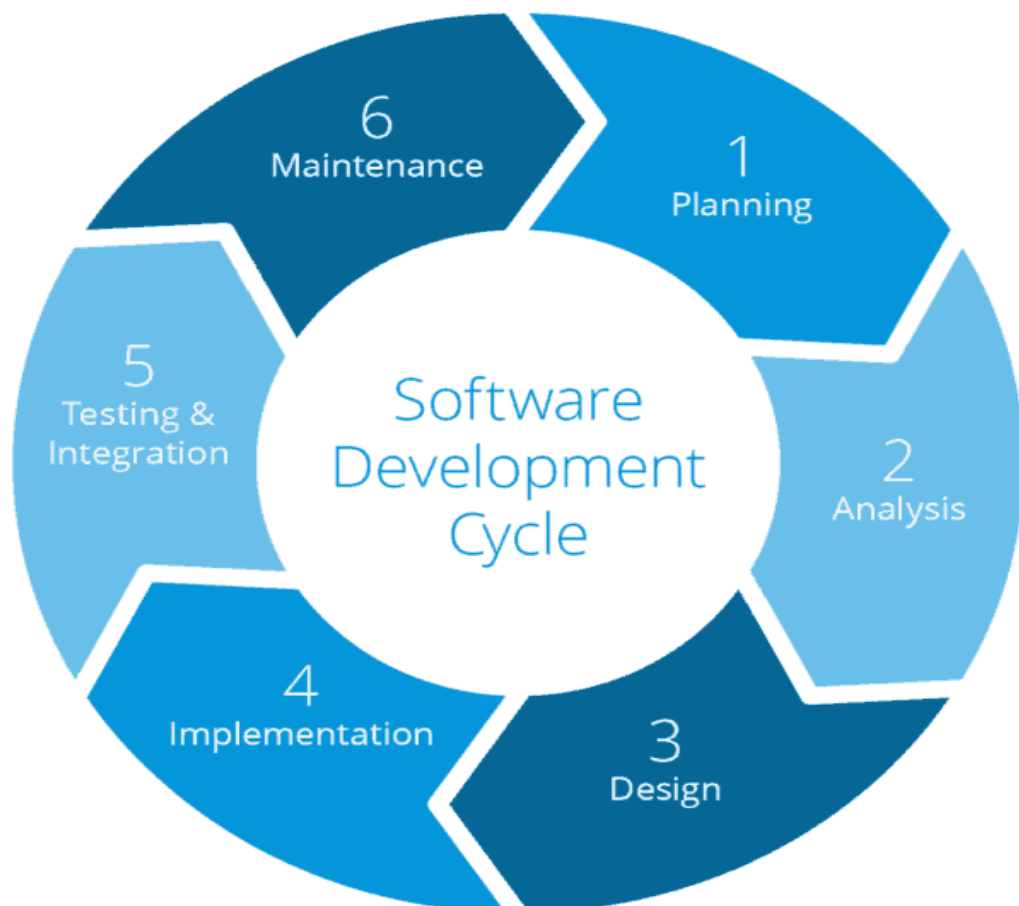


Figure 3.2: Agile Model

I. Planning - The outline of the project is prepared along with the deciding the timeline of the project

II. Analysis - The feasibility of the project and the expected outcome of the project are analyzed

III. Design - The process and workflow of the project are created in this phase

IV. Implementation - Actual programming of the project on the basis of the design laid is carried out

V. Testing and Integration - The complete unit and integration testing of all the modules of the project is performed in this phase

VI. Maintenance - The developed project is now maintained by performing regular checks on the working of the project and making sure that it produces the expected output

Agile model helps to assess the progress of the project at hand at periodic intervals thereby helping to judge the efficiency of the work even at early stages. Agile methodology also helps in avoiding the trickling of errors down the stream, as compared to Waterfall model and other similar models Agile methodology proves to be beneficial with respect to incorporating changes downstream.

Hence Agile methodology will be used during the complete SDLC for the system under consideration.

3.7 SYSTEM IMPLEMENTATION PLAN

This subsection presents a outline of how the system will be implemented efficiently in order to satisfy most of the requirements. This plan includes the following tasks:

- Task 1 : Prepare strategic direction keeping in mind the expected outcome
- Task 2 : Prepare a description or clarity about the scope of the project.
- Task 3 : Conduct the decision making process ensuring active participation of all team members
- Task 4 : Brainstorm for selection of various models feasible for the implementation of the project.
- Task 5 : Process the feedback by end users and mentors and make changes accordingly

CHAPTER 4

SYSTEM DESIGN

System design helps to graphically represent the system as a whole. This chapter discusses the system architecture consisting both bird's eye view architecture and detailed architecture for NLU and NLG.

4.1 SYSTEM ARCHITECTURE

The system architecture represents overall structure of the system. The system takes two inputs namely document database and query from the user and works in two stages, stage one consists of suggesting similar questions from the existing database using NLU and stage two consists of generating relevant questions using NLG.

The system produces output in two stages. In stage system compares the input query with the questions present in the database and suggests relevant questions from the database. This step is accomplished using NLU.

In stage two the system is supposed to generate questions which are not in the database using NLG.

Architecture for the system is discussed as per the points below:

- Bird's Eye View Architecture
- Detailed Architecture
 - NLU Architecture
 - NLG Architecture

4.1.1 BIRD'S EYE VIEW

The diagram represents the bird's eye view of the system:

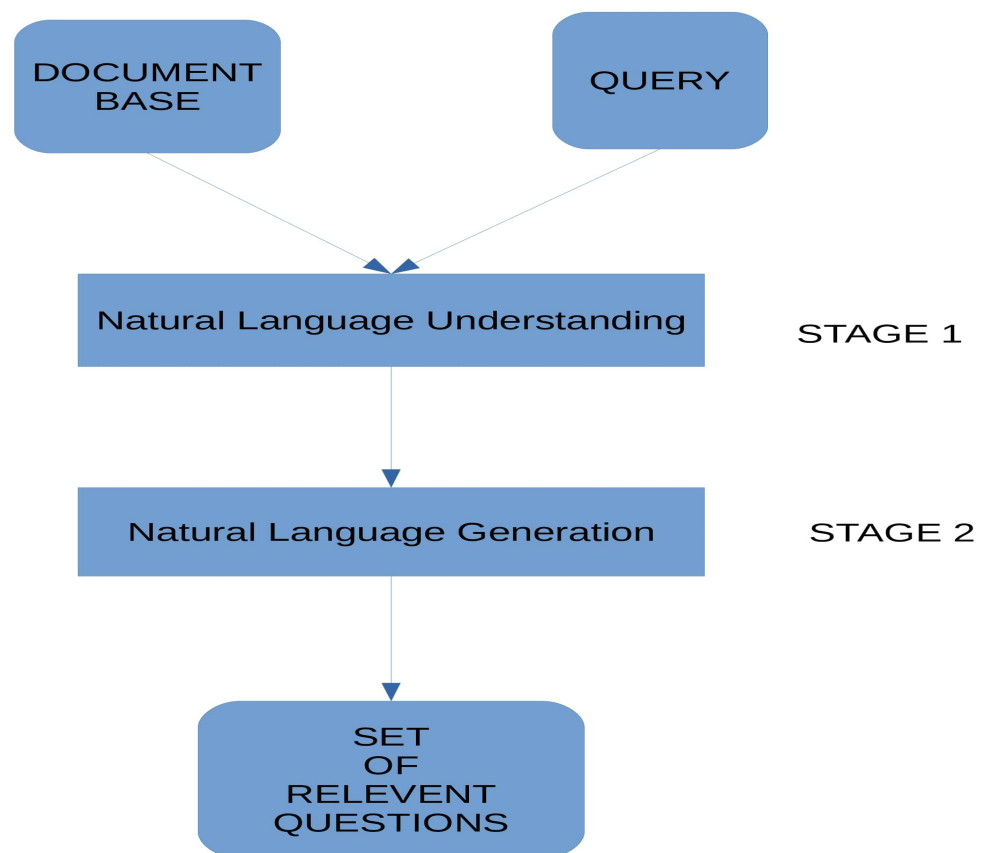


Figure 4.1: Basic Architecture - Overview

4.1.2 NLU DETAILED ARCHITECTURE

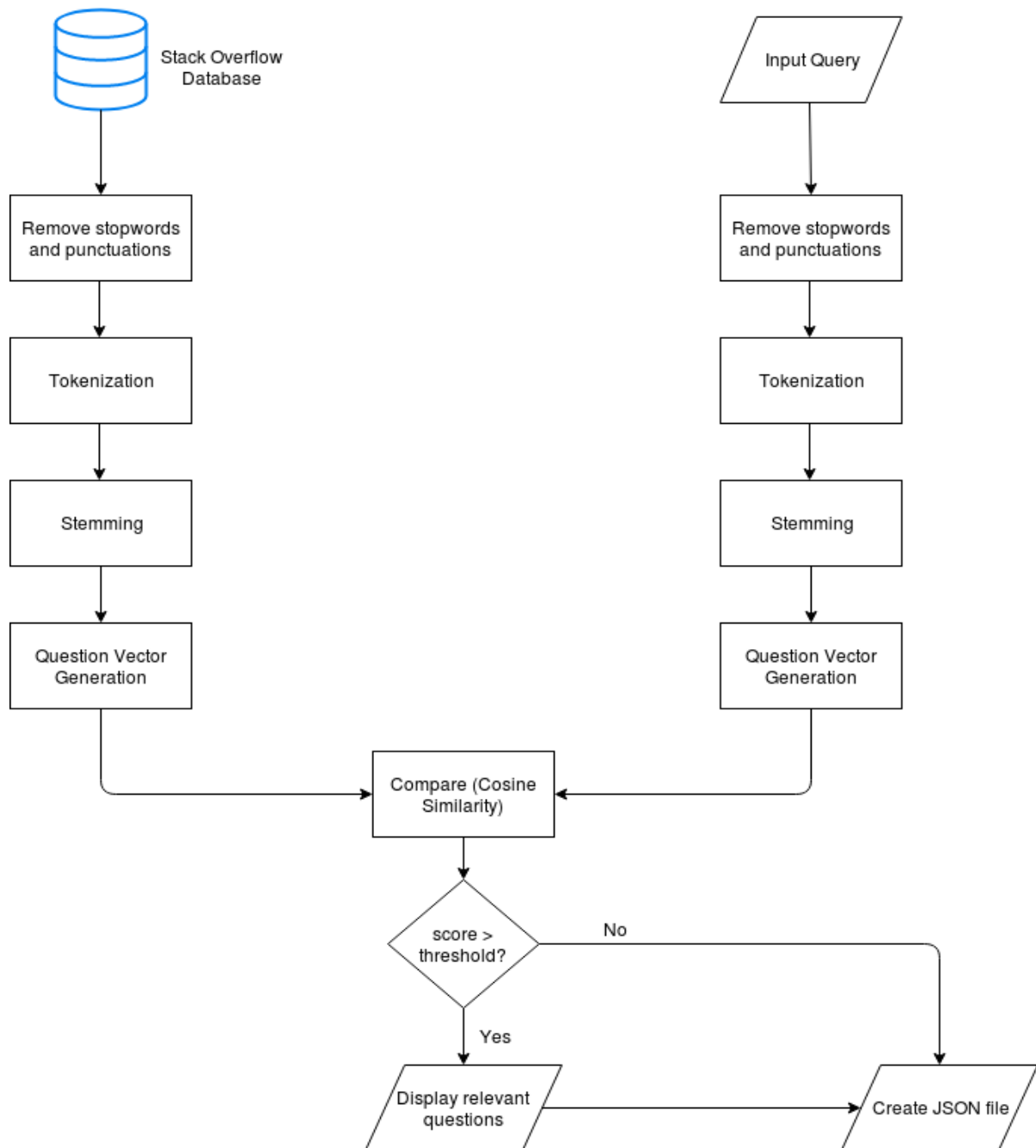


Figure 4.2: NLU Detailed Architecture

4.2 DATA FLOW DIAGRAMS

Data flow diagrams help us to understand the how data is received, managed, stored and transported by the system. DFDs are usually discussed on three different levels, they also happen to be crucial in analysing any pitfalls if there exists any regarding data security in the sytem.

4.2.1 DFD LEVEL 0

DFD Level 0 diagram represents the abstraction view which shows system as a single process with its relationship with external entities.

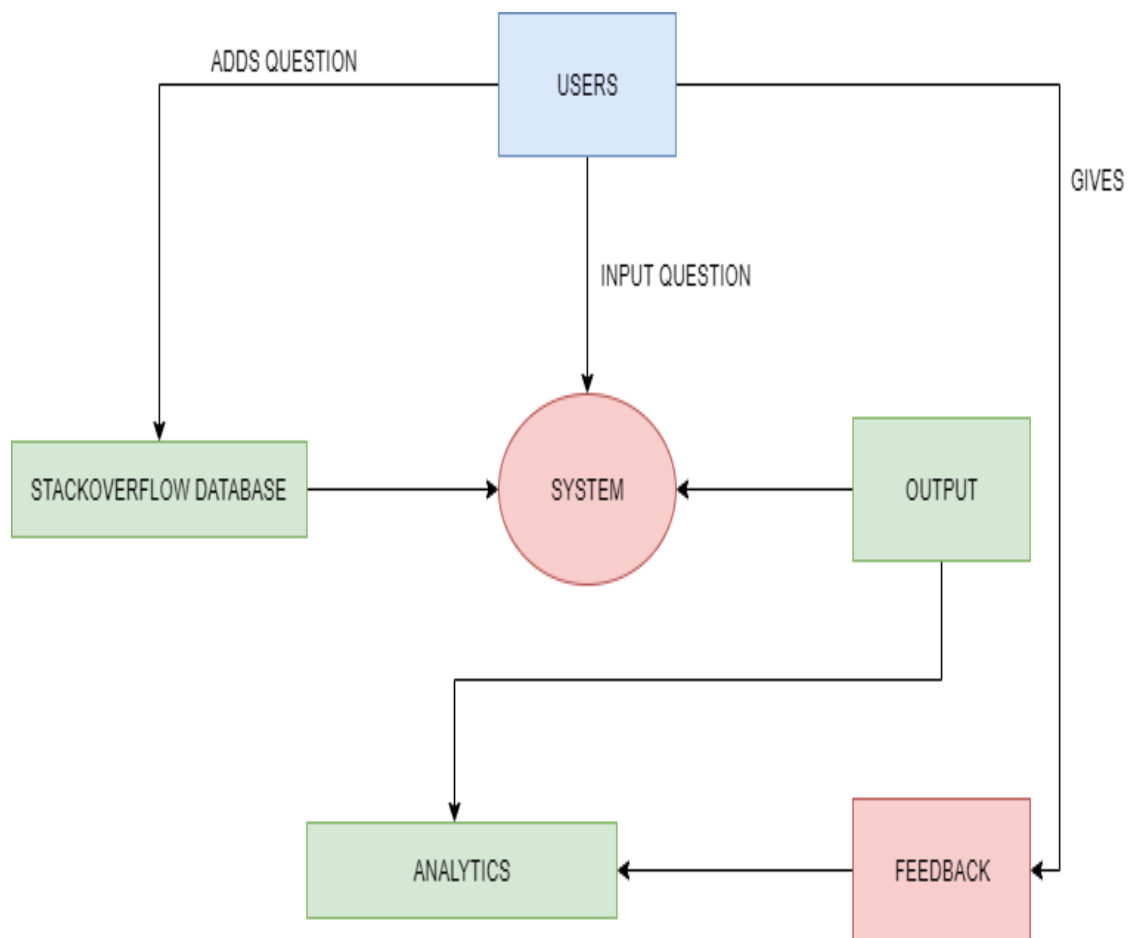


Figure 4.3: DFD Level 0

4.2.2 DFD LEVEL 1

In this level we highlight the main functions of the system and breakdown the high level processes of 0-level DFD into subprocesses.

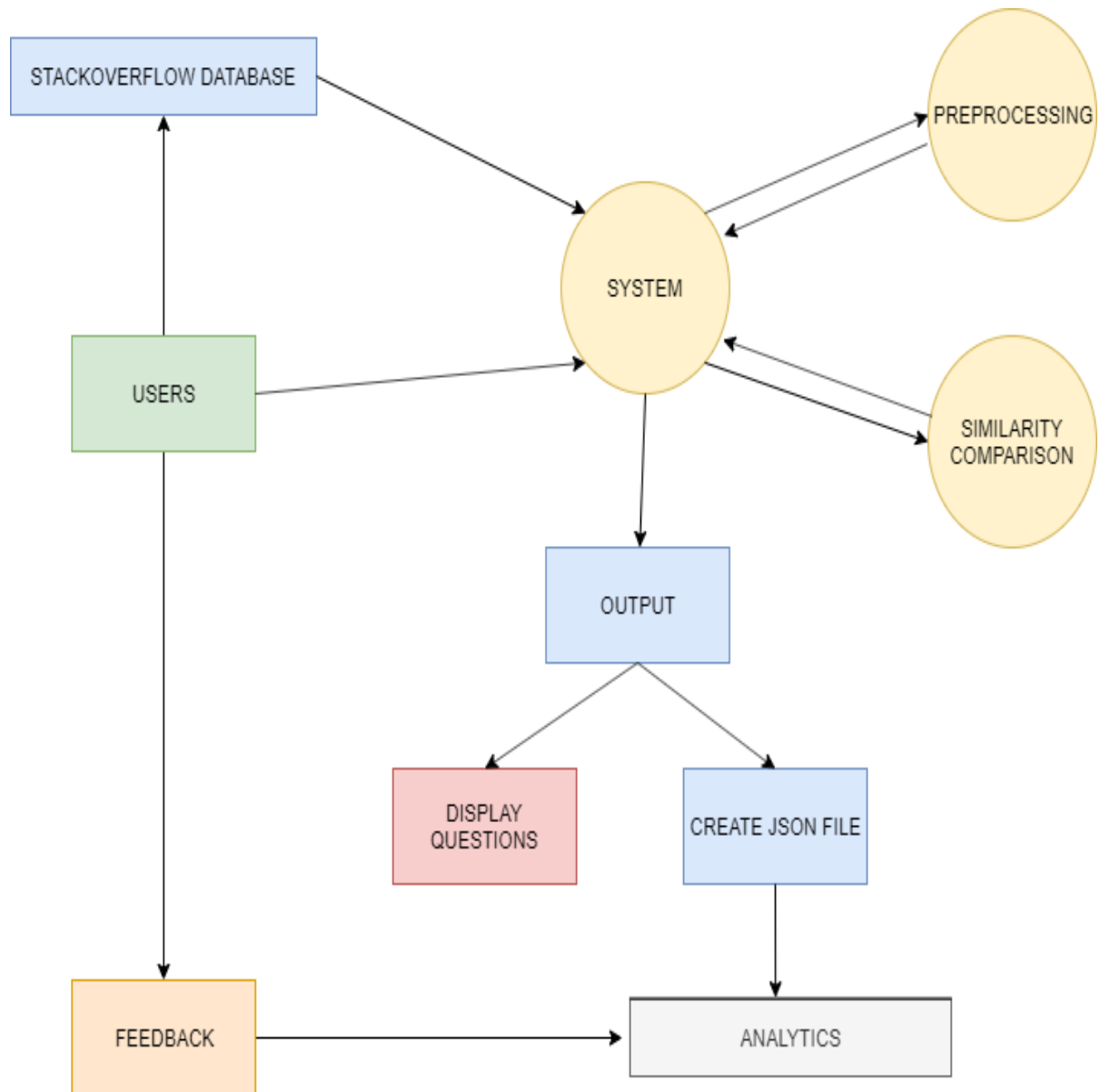


Figure 4.4: DFD Level 1

4.3 UML DIAGRAMS

UML stands for Unified Modelling Language, it is widely used to understand the relationship between the external users and the system components, we can also understand the way users interact with the system using the UML diagram.

4.3.1 USE CASE DIAGRAM

Use case diagram consists of actors and use cases. The actors here are Front End User, System Level User and Administrator.

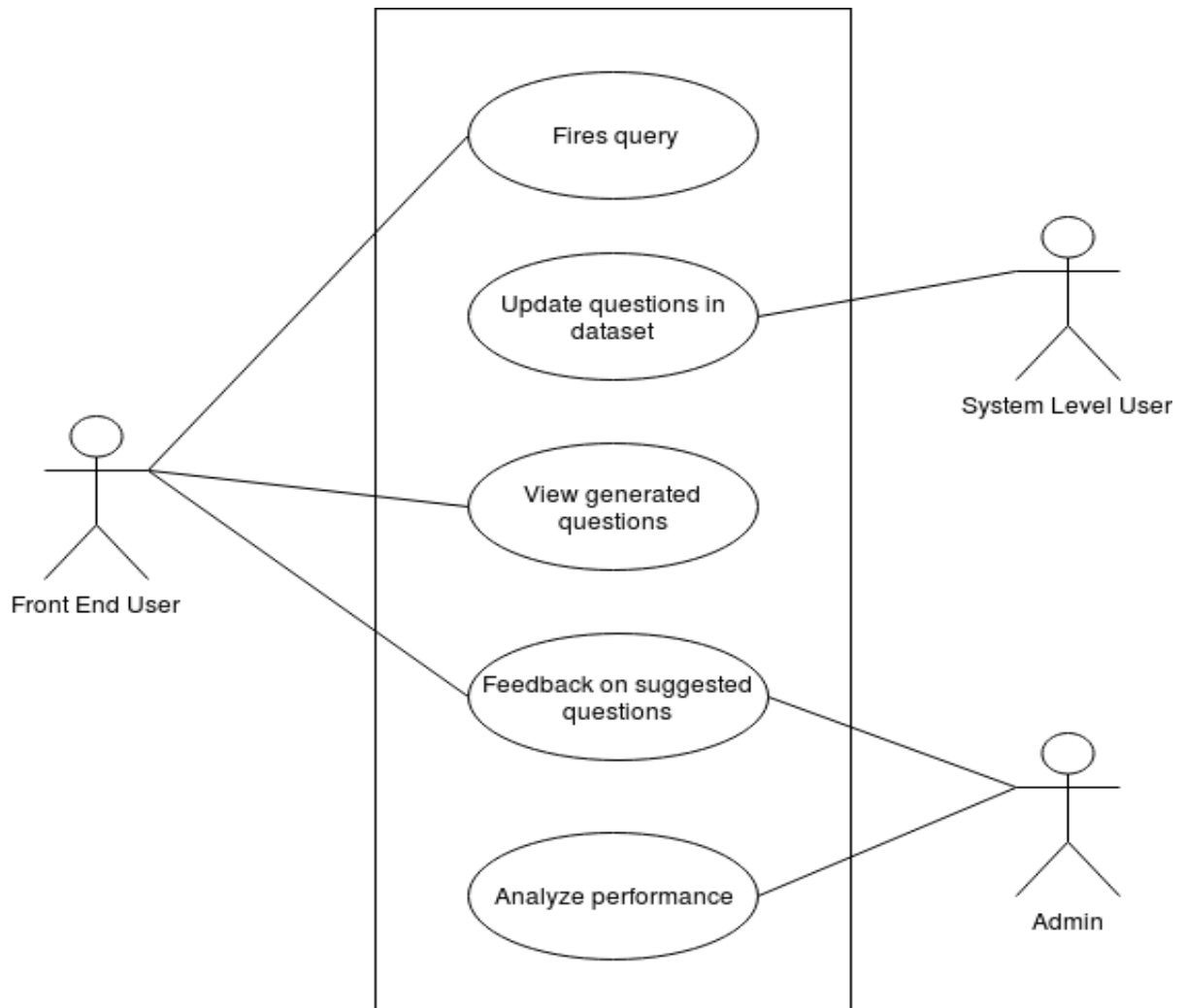


Figure 4.5: Use Case Diagram

CHAPTER 5

OTHER SPECIFICATIONS

5.1 ADVANTAGES

- Less searching required i.e. as the questions are suggested, the user does not have to search extravagantly
- Quick access to most relevant information
- Database or dataset can be built to be retrieved later

5.2 LIMITATIONS

- Rich dataset with many including diverse fields is difficult to construct
- It is difficult to create the same system for indic languages
- The system cannot handle grammatical errors present in the queries

5.3 APPLICATIONS

- Determining questions to be asked in an investigation like a crime investigation
- Determining most relevant citations in a research paper
- Voice based searches - finding most relevant historical news in speech

CHAPTER 6

BASIC IMPLEMENTATION AND RESULT

Basic implementation of the NLU part of the system has been discussed in this chapter, system using which the results were collected has the following specifications:

1. X86-64 Architecture
2. Intel i5-7200 CPU
3. 512 GB SSD, 8GB RAM
4. Python 3.6.9
5. NLTK Library 3.4.5

6.1 ALGORITHM

The code written to implement the below mentioned algorithm has been implemented in python, some libraries used are nltk, matplotlib and others.

Step I Read data from stackoverflow database

Step II Preprocess each data point present in the dataset

Step II.I Remove stopwords and punctuation from the question

Step II.II Tokenize the data and create a word vector representation

Step II.III Performing Stemming on the word vector (Porter Stemmer used)

Step III Read the Input Query and perform step II for the read data

Step IV For every vector generated from the data set

Step IV.I Calculate similarity score using cosine similarity

Step V Display results with similarity score above predefined threshold

Step VI Create json file with all attributes of data and generated similarity threshold

6.2 RESULT

The results obtained include:

- Suggested Questions
- Analysis Json file
- Frequency count of similarity scores generated

The following images illustrate the same:

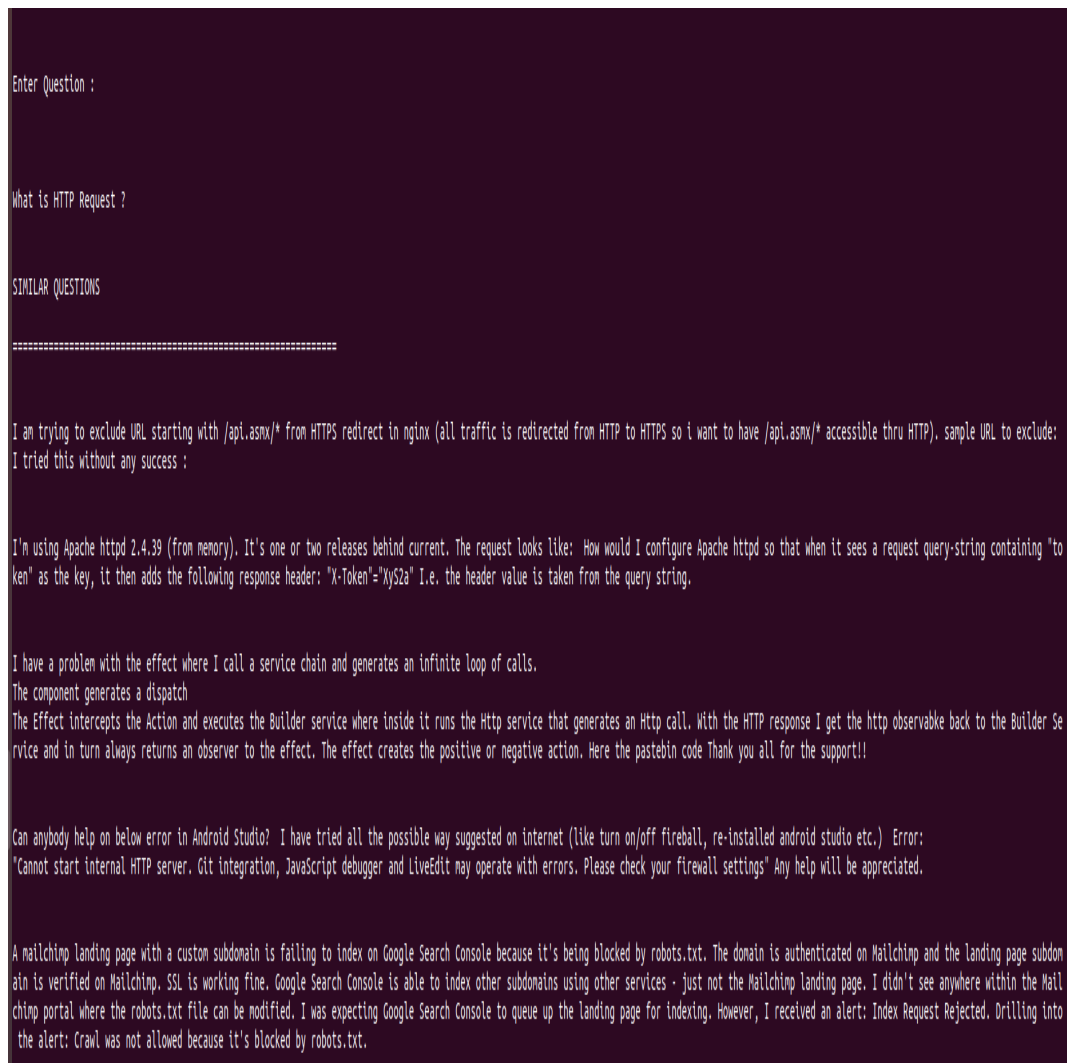
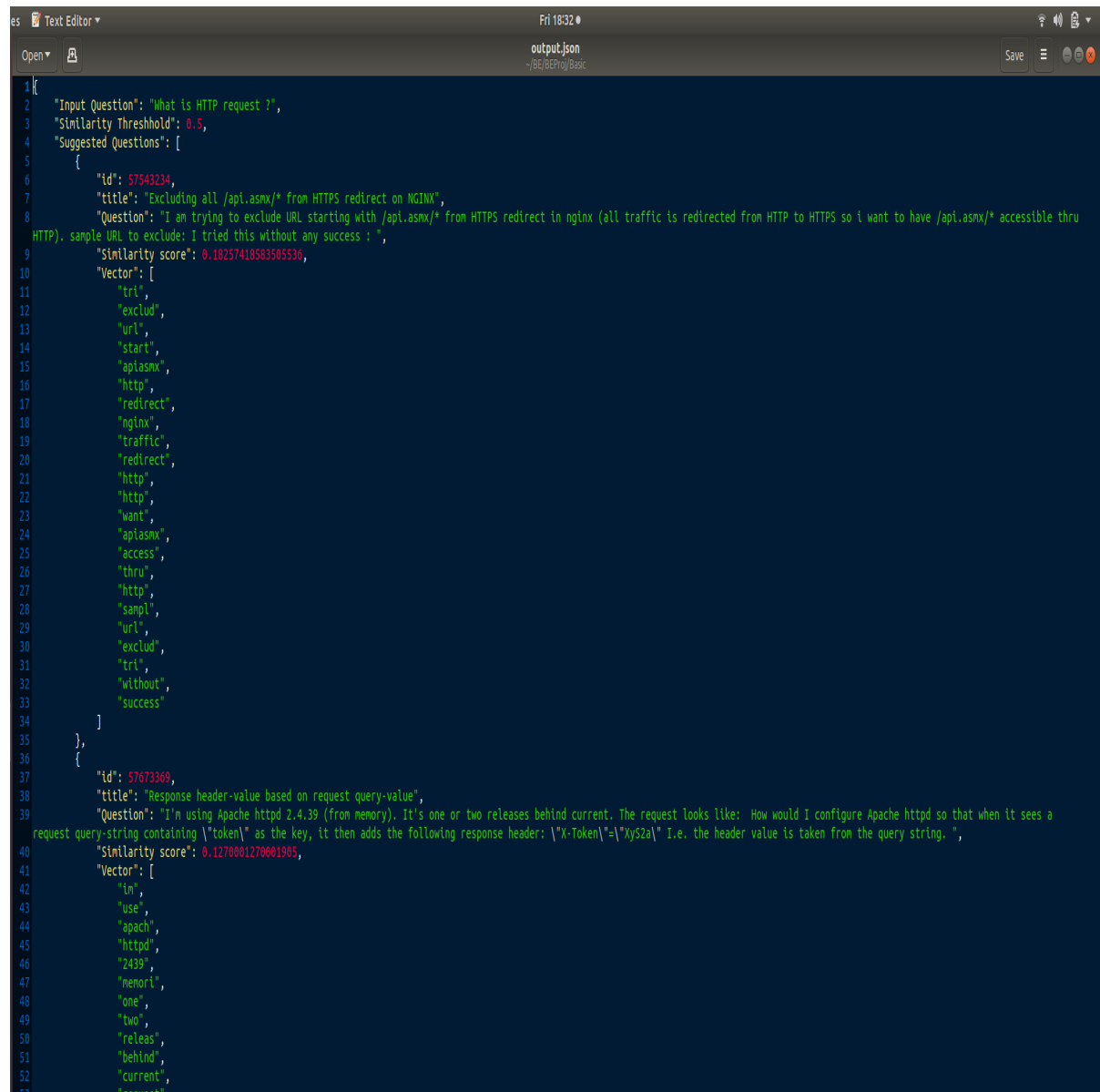


Figure 6.1: Suggested Questions

Generating Relevant Questions For A Query Using Natural Language Processing Techniques.



```
1 {
2   "Input Question": "What is HTTP request ?",
3   "Similarity Threshold": 0.5,
4   "Suggested Questions": [
5     {
6       "id": 57543234,
7       "title": "Excluding all /api.asmx/* from HTTPS redirect on NGINX",
8       "Question": "I am trying to exclude URL starting with /api.asmx/* from HTTPS redirect in nginx (all traffic is redirected from HTTP to HTTPS so i want to have /api.asmx/* accessible thru
9       HTTP). sample URL to exclude: I tried this without any success : ",
10      "Similarity score": 0.18257418583505536,
11      "Vector": [
12        "tri",
13        "exclud",
14        "url",
15        "start",
16        "apiasnx",
17        "http",
18        "redirect",
19        "nginx",
20        "traffic",
21        "redirect",
22        "http",
23        "http",
24        "want",
25        "apiasnx",
26        "access",
27        "thru",
28        "http",
29        "sampl",
30        "url",
31        "exclud",
32        "tri",
33        "without",
34        "success"
35      ],
36    },
37    {
38      "id": 57673369,
39      "title": "Response header-value based on request query-value",
40      "Question": "I'm using Apache httpd 2.4.39 (from memory). It's one or two releases behind current. The request looks like: How would I configure Apache httpd so that when it sees a
41      request query-string containing \"token\" as the key, it then adds the following response header: \"X-Token\"=\"Xy52a\" I.e. the header value is taken from the query string. ",
42      "Similarity score": 0.1278001278001985,
43      "Vector": [
44        "in",
45        "use",
46        "apach",
47        "httpd",
48        "2439",
49        "memori",
50        "one",
51        "two",
52        "releas",
53        "behind",
54        "current",
55        "request"
```

Figure 6.2: JSON File

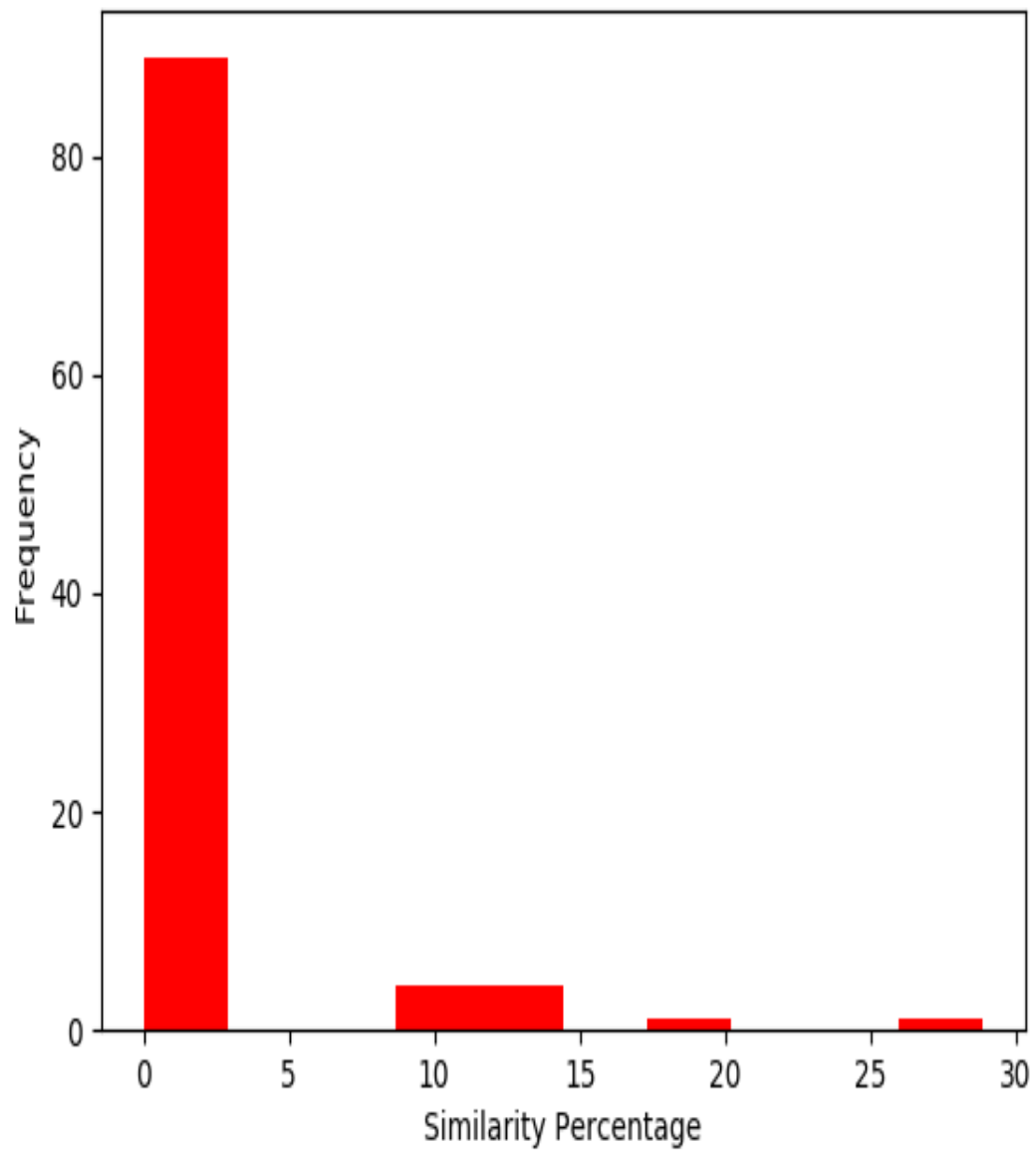


Figure 6.3: Frequency Count of Similarity Index

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

Conclusion states the outcome of the preliminary project report presented on "Generating relevant questions for a query using NLP techniques". This chapter also includes the future scope of the project helping any individual or team taking up the same topic in future.

7.1 CONCLUSION

We have successfully presented a preliminary project report presented on "Generating relevant questions for a query using NLP techniques", illustrating in detail the system requirement, system implementation plan and other required documents that can be referred as a guide during the implementation of the system.

Also, it can be concluded from the basic implementation of stage one of the project that the system successfully suggests similar questions present in the database.

7.2 FUTURE SCOPE

The system can be extended to provide advanced functionalities once the basic requirements are fulfilled. The system capacity can be enhanced using advanced algorithms and techniques. The future scope of the system can be:

- Use of Quantum Computing
- Using Indic NLP in the System
- Creating dynamic graphics as per the question

APPENDIX A

FEASIBILITY STUDY

A.1 PROBLEM STATEMENT FEASIBILITY

This section presents the system using a mathematical model and also discusses the feasibility of the problem. The system analyses if the model suggested belongs to the class of P, NPC or NPH.

The system can be represented as set of six tuple as follows:

$$\begin{aligned} S &= (I, D, F, P, I', D') \text{ where } I = \{x : \text{input query}\} \\ D &= \{y \mid y \in \text{Database}\} \\ I' &= P(I) \\ D' &= P(D) \\ P &= I \rightarrow I' \text{ \& } D \rightarrow D' (\text{PreprocessingFunction}) \\ F &= \{I, D\} \rightarrow R^+ (\text{Function that calculates the similarity threshold}) \end{aligned}$$

The function F used to find similarity works in $O(n)$ complexity where n is the number of words in the sentence, the Pre processing function contains these sub functions:

- Punctuation Removal : $O(n)$
- Tokenization : $O(n)$
- Stemming : $O(k)$

The each of the pre processing functions runs $\forall x \in D$

$$\text{Overall Complexity} = n * (2n + n + n + K) = O(n^2) \quad (1.1)$$

Clearly the complexity can be expressed as a polynomial function of the size of the input, hence the problem of finding similar questions using NLU lies in the class of Polynomial Time Problems.

Appendix B

Details of Papers

B.1 Details Of Papers

[1] In this paper we study about ESA and word2vec similarity techniques. It also states that combining yields better result than individual techniques.

[2] In this paper we calculate similarity using Tree Edit Distance. It gives better accuracy in text similarity.

[3] In this paper we study about the sentence based and phrase based similarity. We also learn about term alignment algorithm.

[4] This is a study paper which helps us to understand various different types of similarity such as Corpus based, Knowledge based etc.

[5] This paper deals with a in depth study about two important text similarity approaches namely Corpus based and Knowledge based text similarity.

[6] In this paper we study about Natural Language generation and we also study about the Semantic based natural Language Generation.

Appendix C

Plagiarism Report

PLAGIARISM SCAN REPORT

Words 931 Date December 07,2019

Characters 6092 Exclude Url

0% Plagiarism	100% Unique	0 Plagiarized Sentences	38 Unique Sentences
------------------	----------------	-------------------------------	------------------------

Content Checked For Plagiarism

According to stack exchange the parent site of stack overflow boasts of 10million views daily. All these views are for catering to their immediate requirement which stack overflow is available to fulfill but fails to develop the overall concept of the user.

1.1 MOTIVATION In today's technologically advancing world, getting answers for questions, general or specific is very important to the development of the user and as a result, development of the overall community eventually. Thus, we propose to build a system to find the relevant questions to a query entered by the user to solve doubts thereby enhancing knowledge of the user. Using text similarity, we can find out relevant questions to the query put forth by the user. One of the most popular sources for people to find answers to their questions is StackOverflow. On StackOverflow people can easily find answers to the questions they have, but there is no suggestion for more questions on StackOverflow. We only get the answers to the question which we have asked, but if more similar questions are suggested this would generate inquisitiveness in the user regarding the subject. This would ultimately help in increasing the knowledge of users by going through relevant question chain.

1.2 PROBLEM DEFINITION Many technology enthusiasts including professionals working in the industry, researchers and students from all over the world use stack overflow as a ready reckoner for almost any difficulty they face in the technical field, while this proves extremely beneficial at the moment but does not help to develop the concept as a whole. Use of stack overflow solves the problem temporarily but does not help in enhancing knowledge. The system to be built aims at providing relevant questions to an input query or question from a database and also generate questions/queries that may prove to be beneficial to the user. The system can be used across many sectors and will prove beneficial in learning, improving profits and reducing time of exploratory tasks by providing help to the user. The system must satisfy basic functional as well as non-functional requirements as discussed in subsequent text in order to qualify as a useful and appreciable project. Any project or system in today's world has to be secure, user friendly and fast enough to survive in the market or to produce itself as a viable option.

3.1.1 PROJECT SCOPE This project is currently being built with stackoverflow database but the system will be able to produce desired when fed with a similar database consisting of set of questions from any field. The project not only aims at understanding the fed query and presenting similar questions from the dataset but also aims at generating questions that will help the user to understand the subject matter at hand better. The project scope pans mainly over presenting and generating queries based on an input user query when presented with a database. The project however does not extend to answering these questions or suggesting any resource to study the same. The project consists of two stages namely NLU (Natural Language Understanding) and NLG (Natural Language Generation). This sub-section discusses the role and privilege level enjoyed by different users of the system. The users of the system can be identified into these broad categories as follows:

- Front End User
- Back End User

The front-end user will be exposed only to the front UI of the system and will be able to query the system and provide feedback based on the output of the system namely the suggested or generated questions. The feedback intended from the front user is predominantly based on the user's judgement of the relevance or usefulness regarding the queries suggested/generated. The Backend user can again be split as follows:

- Admin level user
- System level user

The system level user will be able to upload and modify the database available with the system whereas the admin level user will be able to analyse the performance of the system with the help of the feedback in addition to the functionalities of system level user. Functional requirements are used to define a function of the system or a component of the system as a relationship between input and output. Functional requirements for the proposed system have been discussed in this section. All functional mainly revolve around these three points:

- Purpose - To generate relevant questions for the entered query
- Input - A query or a question which is to be entered by the user
- Output - Set(s) of questions generated for the entered query

The functional requirements for the system can be stated as follows:

- Suggest relevant Questions from the database as per the input query which has some similarity threshold.
- Also check the similarity threshold for the generated questions.
- Based on the

Figure C.1: Plagiarism Report 1

PLAGIARISM SCAN REPORT

Words 860 Date December 07,2019

Characters 6011 Exclude Url

0% Plagiarism	100% Unique	0 Plagiarized Sentences	38 Unique Sentences
------------------	----------------	-------------------------------	------------------------

Content Checked For Plagiarism

Communication interfaces include email, messaging or network protocols that will be required by the system either to fulfill the requirement or provide a functionality. The system under consideration may be required to send periodical mail or files to system admin containing the reports regarding the functioning of the system. For this system communications interface mainly contains reporting of the system performance reports to the system admin. The reports can be generated in JSON format which happens to be easy to use, transport and work on.

3.4 NON-FUNCTIONAL REQUIREMENTS

Non-Functional Requirement (NFR) defines the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system.

3.4.1 PERFORMANCE REQUIREMENTS

The system should be able to give a quick and efficient response for the entered query. The delay of response, if any, should be only due to lack of computation power and not due to inefficiency of the algorithm.

3.4.2 SAFETY REQUIREMENTS

System should not crash due to overabundance of queries. Abrupt failure of system should not happen.

3.4.3 SECURITY REQUIREMENTS

System should maintain a threshold of queries to be handled simultaneously so as to avoid getting into aborted state. Addition of question to the database must be available for only those queries entered by the user i.e. direct addition to the database without displaying similar questions should not happen.

3.4.4 SOFTWARE QUALITY ATTRIBUTES

The software quality attributes for the system under consideration are:

- Maintainability - Need of updation of database whenever query is entered
- Robust - The software should be able to handle multiple queries at a time
- Recoverability - The database should have replicas in case of loss of data
- Availability - User should be able to query the database anytime

3.5 SYSTEM REQUIREMENTS

This section deals with the requirements for the working of the system like Database Requirement, Software Requirements and Hardware Requirements.

- Database Requirements - The database is a BigQuery database, so the Google Cloud Platform (GCP) should be enabled for querying the database if required.
- Hardware Requirements - Intel i5 Processor - GPU for performing computations on large datasets
- Software Requirements - NLTK Library - Python 3.

SDLC stands for Software Development of Life Cycle and represents the methodology for efficient implementation that will be followed during completing the entire project. Different models like Waterfall, V-Model, Agile methodology were taken into consideration and Agile methodology has been chosen for the project as it best suits the nature of the project.

Agile model has six phases namely Planning, Analysis, Design, Implementation, Testing and Integration and Maintenance.

- Planning** - The outline of the project is prepared along with the deciding the timeline of the project.
- Analysis** - The feasibility of the project and the expected outcome of the project are analyzed.
- Design** - The process and workflow of the project are created in this phase.
- Implementation** - Actual programming of the project on the basis of the design laid is carried out.
- Testing and Integration** - The complete unit and integration testing of all the modules of the project is performed in this phase.
- Maintenance** - The developed project is now maintained by performing regular checks on the working of the project and making sure that it produces the expected output.

Agile model helps to assess the progress of the project at hand at periodic intervals thereby helping to judge the efficiency of the work even at early stages. Agile methodology also helps in avoiding the trickling of errors down the stream, as compared to Waterfall model and other similar models. Agile methodology proved to be beneficial with respect to incorporating changes downstream. Hence Agile methodology will be used during the complete SDLC for the system under consideration. This subsection presents an outline of how the system will be implemented efficiently in order to satisfy most of the requirements. This plan includes the following tasks:

- Task 1 : Prepare strategic direction keeping in mind the expected outcome
- Task 2 : Prepare a description or clarity about the scope of the project
- Task 3 : Conduct the decision making process ensuring active participation of all team members
- Task 4 : Brainstorm for selection of various models feasible for the implementation of the project
- Task 5 : Process the feedback by end users and managers and make changes accordingly

The system architecture represents overall structure of the system. The system

Figure C.2: Plagiarism Report 2

PLAGIARISM SCAN REPORT

Words 429 Date December 07,2019

Characters 3145 Exclude Url

0%	100%	0	16
Plagiarism	Unique	Plagiarized Sentences	Unique Sentences

Content Checked For Plagiarism

Less searching required i.e. as the questions are suggested, the user does not have to search extravagantly. Quick access to most relevant information. Database or dataset can be built to be retrieved later.

5.2 LIMITATIONS

- Rich dataset with many including diverse fields is difficult to construct.
- It is difficult to create the same system for indic languages.
- The system cannot handle grammatical errors present in the queries.

5.3 APPLICATIONS

- Determining questions to be asked in an investigation like a crime investigation.
- Determining most relevant citations in a research paper.
- Voice based searches - finding most relevant historical news in speech.

Basic implementation of the NLU part of the system has been discussed in this chapter, system using which the results were collected has the following specifications:

1. X86-64 Architecture
2. Intel i5-7200 CPU
3. 512 GB SSD, 8GB RAM
4. Python 3.6.95, NLTK Library 3.4.56

6.1 ALGORITHM

The code written to implement the below mentioned algorithm has been implemented in python, some libraries used are nltk, matplotlib and others.

Step I Read data from stackoverflow database

Step II Preprocess each data point present in the dataset

Step II.I Remove stopwords and punctuation from the question

Step II.II Tokenize the data and create a word vector representation

Step II.III Performing Stemming on the word vector (Porter Stemmer used)

Step III Read the Input Query and perform step II for the read data

Step IV For every vector generated from the data set

Step IV.I Calculate similarity score using cosine similarity

Step V Display results with similarity score above predefined threshold

Step VI Create json file with all attributes of data and generated similarity threshold

PICT, Department of Computer Engineering 2019-20

Page 26 Generating Relevant Questions For A Query Using Natural Language Processing Techniques.

6.2 RESULT

The results obtained include:

- Suggested Questions
- Analysis Json file
- Frequency count of similarity scores generated

The following images illustrate the same:

Conclusion states the outcome of the preliminary project report presented on "Generating relevant questions for a query using NLP techniques". This chapter also includes the future scope of the project helping any individual or team taking up the same topic in future.

7.1 CONCLUSION

We have successfully presented a preliminary project report presented on "Generating relevant questions for a query using NLP techniques", illustrating in detail the system requirement, system implementation plan and other required documents that can be referred as a guide during the implementation of the system. Also, it can be concluded from the basic implementation of stage one of the project that the system successfully suggests similar questions present in the database.

7.2 FUTURE SCOPE

The system can be extended to provide advanced functionalities once the basic requirements are fulfilled. The system capacity can be enhanced using advanced algorithms and techniques. The future scope of the system can be:

- Use of Quantum Computing
- Using Indic NLP in the System
- Creating dynamic graphics as per the question

Sources

Similarity

Figure C.3: Plagiarism Report 3

Chapter 8

REFERENCES

[1] [2] [3] [4] [5] [6] [7]

- [1] Y. Song and D. Roth, “Unsupervised sparse vector densification for short text similarity,” in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 1275–1280.
- [2] G. Sidorov, H. Gómez-Adorno, I. Markov, D. Pinto, and N. Loya, “Computing text similarity using tree edit distance,” in *2015 Annual Conference of the North American Fuzzy Information Processing Society (NAFIPS) held jointly with 2015 5th World Conference on Soft Computing (WConSC)*. IEEE, 2015, pp. 1–4.
- [3] C. Schwarz, “lsemantica: A command for text similarity based on latent semantic analysis,” *The Stata Journal*, vol. 19, no. 1, pp. 129–142, 2019.
- [4] R. Mihalcea, C. Corley, C. Strapparava *et al.*, “Corpus-based and knowledge-based measures of text semantic similarity,” in *Aaai*, vol. 6, no. 2006, 2006, pp. 775–780.
- [5] A. Kashyap, L. Han, R. Yus, J. Sleeman, T. Satyapanich, S. Gandhi, and T. Finin, “Robust semantic text similarity using lsa, machine learning, and linguistic resources,” *Language Resources and Evaluation*, vol. 50, no. 1, pp. 125–161, 2016.
- [6] A. Islam and D. Inkpen, “Semantic text similarity using corpus-based word similarity and string similarity,” *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 2, no. 2, p. 10, 2008.
- [7] W. H. Gomaa and A. A. Fahmy, “A survey of text similarity approaches,” *International Journal of Computer Applications*, vol. 68, no. 13, pp. 13–18, 2013.